Using a NoSQL graph oriented database to store accessible transport routes

Belén Vela
Escuela Técnica Superior de Ingeniería Informática
Rey Juan Carlos University
28933 Móstoles, Spain
belen.vela@urjc.es

José María Cavero
Escuela Técnica Superior de Ingeniería Informática
Rey Juan Carlos University
28933 Móstoles, Spain
josemaria.cavero@urjc.es

Paloma Cáceres
Escuela Técnica Superior de Ingeniería Informática
Rey Juan Carlos University
28933 Móstoles, Spain
paloma.caceres@urjc.es

Almudena Sierra
Escuela Técnica Superior de Ingeniería Informática
Rey Juan Carlos University
28933 Móstoles, Spain
almudena.sierra@urjc.es

Carlos E. Cuesta
Escuela Técnica Superior de Ingeniería Informática
Rey Juan Carlos University
28933 Móstoles, Spain
carlos.cuesta@urjc.es

ABSTRACT

Each day, people have to move to carry out their daily tasks, such as going to work, studying, shopping, etc., signifying that thousands of trips are taken on public transport on a daily basis. A huge number of these trips are taken by people with special mobility needs. In spite of the existence of numerous Websites and apps that provide information about public transport services, there is a lack of information regarding the accessibility of the routes and sites. We are, therefore, working on the development of a technological framework for the processing, management and exploitation of open data, with the goal of promoting accessibility to city public transport within the framework of the Access@city project. In this paper we specifically focus on the design and storage of accessible transport routes, obtained by means of crowdsourcing techniques, in a NoSQL graph oriented database.

1. INTRODUCTION

According to the World Bank [16], one billion people, or 15% of the world's population, have some type of disability. Although this depends on each country, a significant percentage of the people who use public transport have *special mobility needs.* One of the goals of smart cities is to improve the quality of life of all citizens [12]. In fact, in a smart city, anyone should be able to move easily and according to their needs. There are, therefore, several initiatives whose objective is to improve accessibility to public transport for people with disabilities. For example, the World Health Organization in its "World Report on Disability 2011" [17] proposes to improve accessibility to public transport for people with disabilities, and this includes "making public transport systems more flexible for the user by optimizing the use of information technology".

Various projects and software tools address the issue of public transport and its accessibility. We have carried out a large-scale study of websites and mobile applications that offer information regarding the accessibility of public transport.

The principal eventual aim of our study was to discover the strengths or weaknesses of the public transport information provided and the services offered.

With regard to **public transport *users***, we have analysed the quality and quantity of accessibility information and services; in this case, we have defined six accessibility levels according to the accessibility features related to users' mobility, visual, audible needs, along with other user needs, and the capacity to provide accessible routes related to those user needs, in addition to assigning an accessibility level to each of the applications studied.

With regard to **public transport data**, in addition to identifying the accessibility information contained in them, we have also identified their format in order to determine whether the data provided can be simply managed and reused, thus facilitating their extraction from the Internet and their subsequent use.

All of the websites and mobile applications analyzed provide maps and services and some type of accessibility information, but none of them provides generic mechanisms with which to attain accessible transportation data and which would improve mobility in a smart city. For example, the website accessible.net shows maps with accessibility information, but does not include search options. The website for disabled people, www.discapnet.es, presents information about training, education, employment, legislation, documentation, organization and related services, and includes guides for accessible transport with the option of searching for routes. There are also websites that provide information regarding accessibility for wheelchair users, such as wheelmap.org and Rollstuhlrouting.de. The abil.io website provides information regarding accessible journeys and service-based routing using public transport. The main reason for this is that there is a significant lack of open and reusable data concerning public transport and its accessibility.

In order to address this lack of open transport data and of information regarding accessibility, we are defining an open data repository for accessible public transport within the framework of the Access@City project. The repository will be developed using a NoSQL database owing to its capacity to manage huge volumes of information, along with its flexibility and scalability [14]. We have specifically selected a NoSQL graph oriented database as we are

dealing with highly connected data and wish to be able to make queries that are more efficient in a graph oriented database [7].

We propose to develop the graph oriented database, which will be designed from scratch, by using a methodological approach. In general, we have detected a lack of specific methodologies for the design of NoSQL databases that take into account the application characteristics and the most frequent data queries, which is a particularly important aspect in this kind of systems.

A trend concerning how to incorporate traditional modeling notions in this context has recently emerged [2]. For example, in [7], Kaur and Rani show an example of NoSQL database design. They use an Entity Relationship Model [4] to obtain a conceptual representation of the model and different data models for each NoSQL database (for example, a class diagram for a document database). Buggiotti et al. propose a methodology based on an abstract data model for NoSQL databases called NoAM (NoSQL Abstract Model) [3].

In summary, it could be said that different approaches exist but that no solution has, as yet, been commonly accepted. In our opinion, the key concept here is neither the model nor the representation techniques to be used, but rather the design process and the aspects to be considered. The characteristics of NoSQL databases are different in nature from those of SQL databases. Denormalization and queries must be taken into account from the beginning of the process.

In order to address this lack, in our previous work [15], we proposed some guidelines for the design of document databases in which we integrated the final use of the data and the most frequent queries into the design process.

In this paper, we shall show how to design a NoSQL graph oriented database in which to store accessible routes generated by the users of a mobile application. The accessible routes are obtained for users with special needs by using crowdsourcing techniques (micro tasks) [6][8][9]. For the storage of the routes we specifically use what is, according to [13], the most popular graph oriented database i.e., Neo4j [11].

The remainder of the paper is organized as follows: the framework of our work is briefly presented in Section 2. In Section 3 we present our approach for the design of a graph oriented database for the storage of the accessible public transport routes generated. In Section 4 we show a validation of our proposal, along with a brief description of the mobile application developed for the generation of accessible transport routes and their storage in a Neo4j graph oriented database. Finally, our main conclusions and future work are summarized in Section 5.

2. FRAMEWORK

The framework of our paper is the Access@City project, whose objective is to define a technological framework for the processing, management and exploitation of open data with the goal of promoting accessibility to city public transport (see Figure 1). We therefore address the integration of accessibility data derived from three kinds of sources: 1) existing open data, available from Linked Open Data (LOD) initiatives or obtained using the web scraping of non-semantic data sources; 2) private data concerning actual accessible routes, obtained by means of crowdsourcing and provided by users themselves through their mobile devices and also processed using Big Data techniques, integrating both historical and real-time data in a datastore [10] denominated as “REPOSITORY OF ACCESSIBLE ROUTES” shown in Figure 1, and 3) data obtained from already existing traffic sensors, in the context of a smart city.

These data sources will be semantically harmonized, while maintaining their diversity, and will feed an open data management platform, which consists of a repository (data hub) and a service generation layer. This layer will be able to provide access to data consumers, through the automatic generation of customized APIs composed of services adapted to available data, which will be exploited by different applications. In particular, we consider the case of mobile applications, which would make it possible for citizens to obtain accessible routes between two points in a city in real time, and even combine different transport networks. These apps would translate the information regarding our smart city into an accessibility context, thus resulting in the definition of an accessible city.

The case study that we present in the following section of this paper is focused on the marked part of Figure 1, which includes the application that captures the accessible routes obtained and validated by users using crowdsourcing techniques and the Big Data repository (REPOSITORY OF ACCESSIBLE ROUTES) that will store the routes. We consider that a route is accessible if a person with a special need can use it to reach his or her destination.

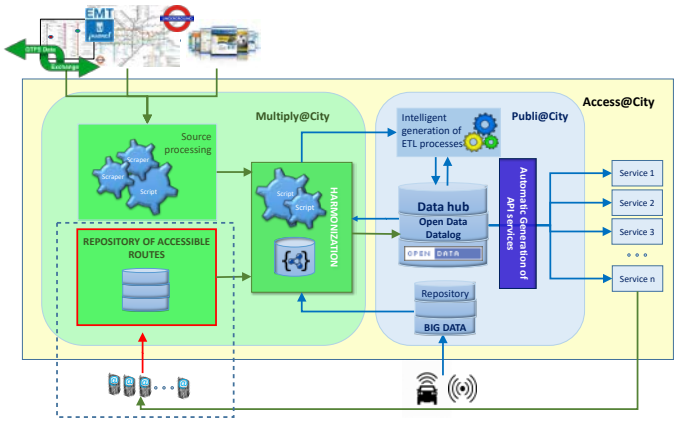

Figure 1. Architecture of Access@City

The remaining parts of Figure 1 show the rest of the architecture on which we are working in the Access@City project.

3. DESIGNING A NOSQL GRAPH ORIENTED DATABASE

Our proposal consists of developing the NoSQL graph oriented database by following a process based on the traditional database design. The proposed approach is summarized in Figure 2 .

In a **first step**, we acquire and analyze the data sources or the specification in order to be able to determine the entities and their relationships, along with their properties. This specification is used to define the conceptual data required to design the conceptual schema of the database from scratch. The conceptual model can be represented using, for example, the Entity-Relationship Model [4] or the UML class diagram [13].

In the **second step**, taking into account the conceptual data model (which is independent of any database technology) and carrying out a study of the application specific access model and the frequent types of queries, we design the logical graph oriented database model, which is independent of any product. This step provides an initial product-independent specification, thus improving the maintainability of the NoSQL database, in addition to making migrations between products easier.

In the **third step**, we attain the physical design and the implementation for a specific NoSQL product, and the product database model is obtained. In our case, we have chosen Neo4j [11], which is, according to the database ranking [5], the most popular graph oriented database. Finally, the implementation phase includes

various physical design tasks, such as balancing the need for scalability, availability, consistency, partition protection and durability.

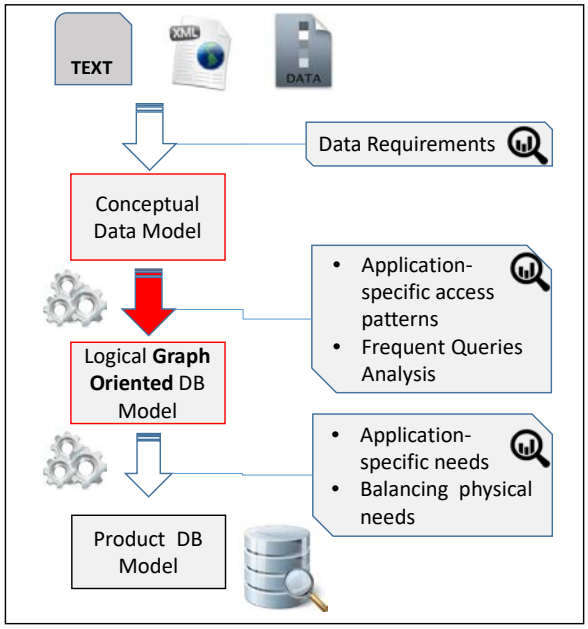

Figure 2. Graph Oriented Database Design Approach

Focusing on the **second step**, that is, on the transformation of the Conceptual Data Model into the Logical Graph Oriented Model (red arrow in Figure 2), we shall begin with a conceptual model represented using the Entity-Relationship (E/R) Model. For the logical graph oriented model, we shall consider "directed graphs", which are graphs composed of nodes (or "vertices") connected by relationships called "edges", each of which is associated with a direction. The direction of the edges is represented by means of an arrowhead on the connecting line between the nodes.

With regard to the transformation, we consider the E/R schema obtained in the first step, the most common queries as regards the data (defined in natural language) and the update operations performed in the database by the applications in an iterative process.

Bearing these aspects in mind, along with the fact that the data can be queried in many ways, we have to decide when to transform an entity or a relationship into a node type or an edge.

In a first iteration, the summarized rules are:

- Each **entity** will be transformed into a **node type** labelled as the entity and its attributes into properties of this node type.

  The **constraints** (uniqueness or not null) of the attributes will be transformed into constraints of the property/ies of a node type.

- Each **relationship** will, in general, be transformed into an edge between the nodes, depending specifically on the cardinality of the relationship.
  - **One-to-one relationships (0/1 to 0/1)** will be transformed into an edge (without an arrowhead) between both node types to denote the **1:1 relation** between the entities.
  - **One-to-many relationships (0/1 to 0/n)** will be transformed into an edge with an arrowhead to denote the **1:N relation** between the entities.
  - **Many-to-many relationships (0/n to 0/m)** will be transformed into an edge between both node types with an arrowhead on each end to denote the **N:M relation** between the entities. At this point, we have to decide and check whether this relationship should be transformed into an edge or a node.
  - A **generalization** is a special kind of relationship and will be transformed in the same way as the other types of relationships, according to its cardinality and including an edge labelled "is-a".
  - A **composition** is a special kind of relationship and will be transformed in the same way as the other types of 1:N relationships, according to its cardinality and including an edge labelled "is-composed-of".

After this first iteration, we have to refine our logical graph oriented DB model, taking into account both the access patterns of the applications and frequent queries in order to be able to query the connected data in many ways, as required by the users.

4. VALIDATION: APPLICATION FOR GENERATING AND STORING ACCESIBLE ROUTES USING A GRAPH ORIENTED DATABASE

In order to validate our proposal, we have developed a native Android application as we need to use the GPS of the users' device. In general, native applications have significant advantages over hybrid applications because they are able to easily access and use the built-in capabilities of the user's devices (e.g., GPS) [1].

This application will allow users to register for the generation of accessible routes. They can then use the starting a route option, indicating which special need (wheelchair, bike, baby stroller, baby buggy, etc…) they will have on their journey. During the journey, the application will periodically register the GPS position (initially, every 25 seconds, although this could change depending on the route, the special need, etc.). When the user finishes the journey, he/she can either discard the route or save it. Users may include comments about the routes taken, reporting possible incidents and/or including photos.

Figure 3 shows the main functionalities of the application by means of a Use Case Diagram:

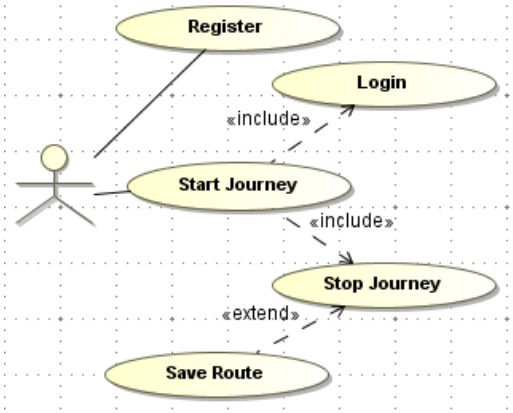

Figure 3. Use Case Diagram

In Figure 4, some of the main screenshots of the application developed are shown.

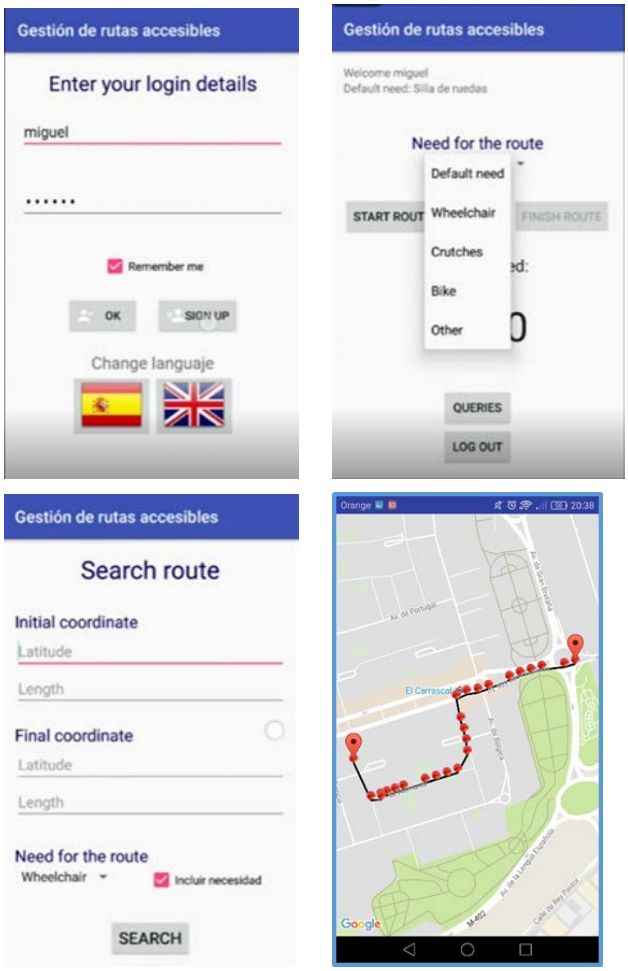

Figure 4. Screenshots of the application "Gestión de rutas accesibles" (Generation and Storage of Accesible Routes)

For the storage of the routes, we have developed a Big Data repository in which to store accessible routes obtained by means of the aforementioned application using crowdsourcing techniques.

The Big Data repository in our Case Study was developed by first identifying the data requirements. We then defined the conceptual data model using an Entity-Relationship Data Model (Figure 5).

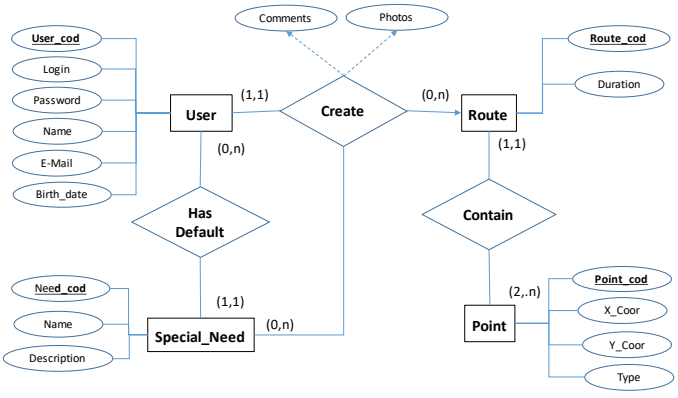

Figure 5. Conceptual Data Model

This **conceptual data model** includes user registration data (USER Entity) and information about the routes (ROUTE Entity) taken by users with special needs (SPECIAL_NEED Entity). We also consider possible comments made and/or photos taken by users on their journeys. The routes are composed of at least two GPS points (POINT Entity). Each of these points is composed of X and Y coordinates and has a Type, which can be "start point", "intermediate point" or "end point"

At this point, another necessary and important decision that had to be made was which kind of NoSQL database to use for the development of our big data repository. In this work, we have chosen a graph oriented database owing to the nature of the data of the routes, which is highly connected, and the need to query the data in many ways. The methodology shown in Figure 2 assumes that the database chosen is a NoSQL graph oriented database. The case study has been implemented in Neo4j. The application will be connected to the Neo4j database using API REST and an HTTP connection.

In a previous work [15], we developed our repository using a NoSQL document database because of its flexibility and ability to manage complex data structures [14]. However, as the data in our case study are highly connected (point of a route), the traversal is much simpler using a graph oriented database.

In order to obtain the **logical graph oriented database model**, in a first iteration we have to apply the proposed guidelines. We then have to consider how the data will be used by the applications, that is, what the most frequent queries and application-specific access models are.

In our case study, the most common queries will be related to users or to routes. The most frequent queries are:

1) Data of the registered users, including their default special need.

2) Is a route between two given points accessible?

3) Comments concerning or photos of the stored accessible routes.

4) Of which points is a route composed?

Apart from the queries, there will also be two basic update operations in the database: inserting a new registered user or inserting a new route.

Bearing in mind the conceptual data model and the aforementioned queries (defined in natural language), we have designed the new model (according to our proposed guidelines):

a) A **node type labelled** for each entity: User, Special_Need, Route and Point. The attributes of the entity become the properties of each node type, as can be seen in Figure 6.

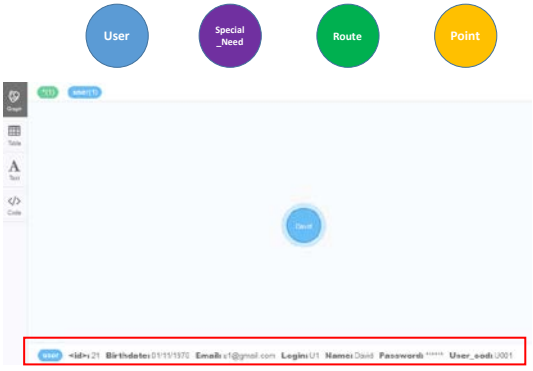

Figure 6. Node of User type with its properties

b) It is now necessary to carry out the **transformation of the existing relationships**. Here, we have to decide whether relationships will be implemented using an edge or a node.

There are three relationships: Has Default relationship (1;N), Create relationship (1:N:M) and Contain relationship (1:N).

c) When **analyzing the queries and the application data**, we consider grouping the information of certain entities of the conceptual data model that will be used together. We therefore exclude the Special_Need node, as its information can be considered as information regarding the User and the Route. We shall, therefore, include the Special Need information as properties of the User and Route node types. This signifies that we now continue working with only three node types and two relationships.

In order to transform both 1:N relationships, we create an edge labelled as the relationship with an arrowhead to denote the 1:N relation between the entities: Create and Contain.

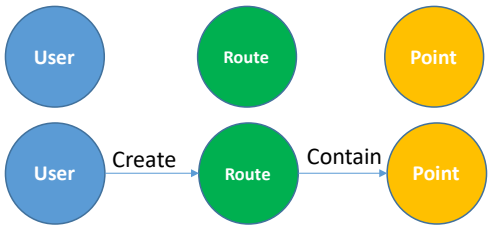

Figure 7. Node types with edges

The final graph oriented database design will, therefore, consist of three node types: USER, ROUTE and POINTs. USERS will store, for each user, a set of information (name, email, birth date, etc…) and information on their default special needs. ROUTES will store the special needs for that route, and some additional information, such as comments, photos, etc. POINTs will store information concerning the X and Y coordinates and the type of point (initial, intermediate and end point). The information regarding routes and subroutes with a special need can be easily obtained with this design.

5. CONCLUSIONS

Although document databases have a dynamic schema (but are not schemaless, as stated in some forums), it is very important to design this schema correctly because of its impact on the performance of the database. A methodological process with which to guide the user in the design process of a NoSQL database is, therefore, required. However, despite the existence of several works and many websites with best practices, there is no commonly accepted solution for the design of NoSQL databases.

In [15], we proposed some guidelines for the design of a document database. In this paper, and in order to complete our NoSQL Design Methodology, we address the design of NoSQL graph oriented databases. In order to validate this proposal, we present a case study in which we generate accessible routes created by users with special mobility needs using a micro-task based on crowdsourcing and store them in a NoSQL graph oriented database in Neo4j. The design process is principally based on the requirements of the applications and the most frequent queries that the system will have to deal with.

One of our future works will be the formal representation of the queries and the specification of a formal method with which to transform the conceptual model and the query model into a logical design model based on a NoSQL database (document DB or graph oriented DB). We plan to put the application into use with users with disabilities in the near future. Moreover, an immediate future task is to extend the functionalities of the mobile application in order to create a version that can be evaluated by users with special needs.

ACKNOWLEDGMENTS

This work was partially supported by the Access@City project (TIN2016-78103-C2-1-R), funded by the Spanish Ministry of Economy and Competitiveness.

REFERENCES

[1] Abed, R. (2016). Mobile. Hybrid vs Native Mobile Apps – The Answer is Clear. Retrieved from: https://ymedialabs.com/hybrid-vs-native-mobile-apps-the-answer-is-clear/

[2] Atzeni, P. (2015) Models for NoSQL databases: a contradiction? Presentation. Sezione di Informatica e Automazione.

[3] Buggioti, F. , Cabibbo, L., Atzeni, P. and Torlone, R. (2014). Database Design for NoSQL Systems. ER 2014, LNCS 8824, pp.223-231.

[4] Chen, Peter (March 1976). The Entity-Relationship Model - Toward a Unified View of Data. ACM Transactions on Database Systems. 1 (1): 9–36.

[5] DB-Engines Ranking (2017). https://db-engines.com/en/ranking.

[6] Estellés Arolas, E., González Ladrón de Guevara, F. (2012). Towards an integrated crowdsourcing definition. Journal of Information Science, 38(2): 189-200.

[7] Frisendal, T. (2016). Graph Data Modeling for NoSQL and SQL. Visualize Structure and Meaning. Technincs Publications.

[8] Grier, D. A. (2013). Crowdsourcing For Dummies. Paperback

[9] Ke Mao , Licia Capra , Mark Harman , Yue Jia - A survey of the use of crowdsourcing in software engineering (2016). The Journal of Systems and Software

[10] Marz, N.; Warren, J. (2015). Big Data: Principles and best practices of scalable realtime systems. Manning.

[11] Neo4j. https://neo4j.com/product/

[12] Neirotti, P., De Marco, A., Cagliano, A. C., Mangano, G., & Scorrano, F. (2014). Current trends in Smart City initiatives: Some stylised facts. Cities, 38, 25-36.

[13] Object Management Group (2015) OMG Unified Modeling Language TM (OMG UML) Version 2.5. Retrieved from http://www.omg.org/spec/UML/2.5/ Solid IT (2017).

[14] Sullivan, D. (2015). NoSQL For Mere Mortals. Addison-Wesley.

[15] Vela, B., Cavero, J.M., Caceres, P., Sierra, A.& Cuesta, C.E. (2017), Defining a NoSQL document of accessible transport routes. Darli-Ap 2017 in iThings-GreenCom-CPSCom-SmartData 2017. Exeter, UK.

[16] World Bank. http://www.worldbank.org/en/topic/disability/overview

[17] World Report on Disability, 2011. http://www.who.int/disabilities/world_report/2011/report/en/